

Séminaire d'Installation d'Arch Linux

Département d'informatique

Eric Hervet et Andy Couturier

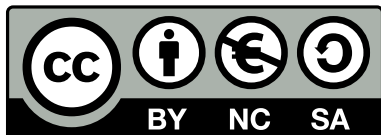
Département d'informatique
Université de Moncton

27 novembre 2024



Table des matières

1. Support d'installation
2. Partitionnement, formatage et montage
3. Installation
4. Chroot
5. Bootloader
6. Initramfs
7. Compression de la mémoire
8. Horloge et fuseau horaire
9. Langue du système
10. Fichier hosts et hostname
11. Comptes utilisateurs
12. Arch User Repository (AUR)
13. Installation d'un bureau graphique
14. Redémarrage



Ce document est mis à disposition selon les termes de la licence **Creative Commons “Attribution – Pas d’utilisation commerciale – Partage dans les mêmes conditions 4.0 International”**.

1. Support d'installation

1

Support d'installation

- Pour créer une clé USB qui contient l'image d'installation d'Arch Linux :
 1. Rendez vous à l'adresse <https://archlinux.org/download/>.
 2. Choisissez un serveur Canadien dans la liste et cliquez sur son lien.
 3. Via la page d'index du serveur, téléchargez le fichier ISO qui aura un nom comme : `archlinux-20XX.XX.01-x86_64.iso`.
- Si vous êtes déjà sur Linux :
 1. Ouvrez un terminal.
 2. Branchez la clé USB qui servira de support d'installation.
 3. `$sudo dmesg` pour identifier le code sdX de la clé USB.
 4. Placez vous dans le répertoire où se trouve l'ISO en utilisant `cd`.
 5. `$sudo dd bs=32M if=archlinux-20XX.XX.01-x86_64.iso of=/dev/sdX conv=fsync oflag=direct status=progress`
- Si vous êtes sur Windows (nos sincères sympathies) : Vous pouvez utiliser [Rufus](#).

2. Partitionnement, formatage et montage

2.1 Table de Partitionnement

- **Définition** : Une table de partitionnement est une structure située sur un disque décrivant comment l'espace de stockage est divisé en zones distinctes appelées partitions.
- **Rôle** : Elle détermine où chaque partition commence et se termine sur le disque et le type de système de fichiers de chaque partition.
- **Fonctionnement** :
 - Placée généralement à un emplacement fixe sur le disque, elle est utilisée par l'ordinateur au démarrage pour détecter et gérer les partitions présentes.
 - Indique quelles partitions sont amorçables (« *bootable* »), aidant ainsi le processus de démarrage du système d'exploitation.
- **Systèmes de fichiers** :
 - Chaque partition peut être formatée avec un système de fichiers spécifique, permettant de stocker et d'organiser les fichiers.
 - Différents systèmes de fichiers peuvent coexister sur le même disque dans différentes partitions.

- Il y a deux formats de table de partitionnement possible.
- **MBR (Master Boot Record)**
 - Taille maximale de disque : 2 TiB.
 - Limité à 4 partitions primaires.
 - Utilise un secteur de démarrage principal (*446 bytes*).
 - Plus ancien, compatible avec BIOS uniquement.
 - Pas de redondance, si le MBR est corrompu, les données peuvent être inaccessibles.
- **GPT (GUID Partition Table)**
 - Taille maximale de disque : 9.4 ZiB.
 - Supporte jusqu'à 128 partitions primaires.
 - Utilise des identifiants globaux uniques (GUID) qui permettent l'identification précise des partitions avec un identifiant PARTUUID.
 - Plus moderne, conçu pour UEFI.
 - Redondance avec deux copies de la table, une au début et une à la fin du disque.
 - ▶ Inclut un hash CRC32 pour détecter la corruption de la table.

2.3

Table de partitionnement

#	Device	Mountpoint	Type	Taille
1	/dev/sdX1	/boot (/mnt/boot)	ef00	1 GiB
2	/dev/sdX2	/ (/mnt)	8300	Reste

- Les mountpoints entre parenthèses seront les mountpoints temporaires des partitions pendant l'installation. Ils se trouveront sur le système de fichier existant du support d'installation.
- Le type de partition `ef00` indique qu'il s'agit d'une partition EFI. Cela signifie que le système à un firmware UEFI pour le démarrage.
- La taille de 1 GiB est généralement suffisante pour stocker le noyau Linux et les fichiers de démarrage nécessaires.
- Le type de partition `8300` est l'identifiant pour une partition Linux standard sous le schéma de partitionnement GPT.
- La partition racine utilisera le reste de l'espace libre du disque.

2.4 Partitionnement

- Branchez la clé sur laquelle vous allez installer Arch Linux et ensuite faites immédiatement `$dmesg` pour identifier le code `sdX` de la clé USB.
- `$gdisk /dev/sdX` pour commencer à partitionner le disque interactivement.
 - `o` pour effacer toutes les partitions.
 - `n` pour créer une nouvelle partition.
 - `Enter` pour choisir la partition par défaut (1).
 - `Enter` pour choisir le secteur de départ par défaut.
 - `+1G` pour choisir le secteur de fin 1 GiB après le secteur de départ.
 - `ef00` pour le type de partition.
 - `n` pour créer une nouvelle partition.
 - `Enter` pour choisir la partition par défaut (2).
 - `Enter` pour choisir le secteur de départ par défaut.
 - `Enter` pour choisir le secteur de fin par défaut (fin du disque).
 - `Enter` pour choisir type de partition par défaut (8300).
 - `p` pour vérifier le partitionnement.
 - `w` puis `y` pour écrire la table de partitions au disque.

2.5

Formatage de la partition de démarrage

- `$mkfs.fat -F32 /dev/sdX1`
- **Conformité UEFI (Unified Extensible Firmware Interface)** : Le standard UEFI spécifie une partition système EFI (ESP) au format FAT32.
- **Pilotes UEFI intégrés** : Les firmwares UEFI incluent des pilotes FAT. Sans ce format, l'UEFI ne peut ni localiser ni charger le bootloader, rendant le démarrage impossible.
- **Compatibilité universelle** : FAT32 est pris en charge nativement par la plupart des systèmes d'exploitation, simplifiant la mise à jour de la partition (fichiers EFI, bootloaders, etc.) depuis n'importe quelle plateforme.

2.6 Formatage de la partition racine

- `$mkfs.f2fs /dev/sdX2`
- **F2FS (Flash-Friendly File System)** a été conçu spécifiquement pour les supports de stockage à base de mémoire flash (SSD, eMMC, clés USB).
- **Log-structured file system** : Au lieu de réécrire directement au même endroit, les données modifiées sont toujours inscrites dans de nouveaux blocs. Cette approche transforme les accès aléatoires en flux d'écritures séquentielles, réduisant la fragmentation et améliorant considérablement les performances.
- **Garbage collection** : L'espace est divisé en segments, et ceux contenant des données obsolètes sont régulièrement regroupés en blocs contigus. Ce processus limite l'amplification d'écriture, maintient un stockage homogène et préserve durablement les performances.
- **TRIM et wear-leveling** : Le TRIM indique au contrôleur flash quels blocs sont libres, lui permettant de répartir uniformément l'usure au niveau des cellules physiques du support (wear-leveling) et ainsi prolonger sa durée de vie.

- `$mount /dev/sdX2 /mnt` : On monte la partition racine dans le système de fichiers temporaire fourni par l'environnement d'installation.
- `$mkdir /mnt/boot` : Création du répertoire `/mnt/boot` pour accueillir la partition de démarrage.
- `$mount /dev/sdX1 /mnt/boot` : On monte la partition de démarrage dans ce répertoire, intégrant ainsi l'arborescence de `/mnt` dans celle du support d'installation, ce qui permet d'accéder à la structure complète du futur système.

3. Installation

3.1

Initialisation du keyring

- L'une des plus grandes forces de Linux réside dans son système de paquets. Chaque paquet est distribué avec une signature cryptographique, garantissant son authenticité et son intégrité.
- Avant d'installer le moindre logiciel, il est donc nécessaire d'initialiser et de peupler un keyring local, afin que pacman puisse vérifier la validité des paquets grâce aux clés des développeurs de confiance.
- `$pacman-key --init` : Initialise la base de clés, créant l'environnement de confiance nécessaire à la vérification des signatures.
- `$pacman-key --populate` : Importe les clés officielles dans le keyring local, assurant que les paquets installés proviennent de sources légitimes.

3.2 Optimisation des miroirs

- Grâce à la vérification cryptographique des paquets, il n'est pas nécessaire de faire confiance au miroir lui-même, puisqu'un paquet non-signé ou altéré sera rejeté automatiquement par pacman.
- Ainsi, n'importe qui peut héberger un miroir des dépôts Arch Linux. Un miroir est une copie du serveur de paquets officiel des développeurs d'Arch.
- L'écosystème compte plus de 800 miroirs à travers le monde, certains sont plus proches, donc ils ont une latence réduite et une bande passante supérieure.
- Avant d'installer le système, il est donc préférable d'optimiser la liste des miroirs pour télécharger les paquets plus rapidement.
- `$pacman --noconfirm -Syy reflector` : Installe et met à jour la base de données des paquets, puis installe `reflector` un outil d'optimisation des miroirs.
- `$reflector --verbose --sort rate --country Canada --protocol http,https --save /etc/pacman.d/mirrorlist` : Met à jour la liste des miroirs en utilisant ceux au Canada et trié du plus rapide au plus lent.

3.3 Optimisation de pacman

- **Accélérer le téléchargement des paquets :**

Pour améliorer les performances de pacman, éditez le fichier de configuration `/etc/pacman.conf`. Décommentez la ligne `ParallelDownloads` puis définissez sa valeur à **32** (ou un autre nombre selon vos préférences).

```
#ParallelDownloads = 5  
ParallelDownloads = 32
```

Cette modification permet d'effectuer plusieurs téléchargements en parallèle, réduisant significativement le temps total de mise à jour ou d'installation de paquets.

3.4 Installation du système

- `$pacstrap /mnt base base-devel linux linux-headers linux-firmware xorg-drivers nvidia-open intel-ucode amd-ucode zram-generator git`
 - **base** : Paquets essentiels pour un système Arch minimal.
 - **base-devel** : Ensemble d'outils de développement pour la compilation de logiciels.
 - **linux** et **linux-headers** : Noyau Linux et ses en-têtes nécessaires pour le développement de modules.
 - **linux-firmware** : Collection de firmwares pour divers matériels pour assurer un bon support matériel.
 - **xorg-drivers** et **nvidia-open** : Pilotes graphiques (Intel, AMD, Nvidia) pour assurer un bon support matériel.
 - **intel-ucode** et **amd-ucode** : Microcodes pour les processeurs Intel et AMD, permettant des mises à jour critiques du microprogramme.
 - **zram-generator** : Utilitaire permettant de configurer zram pour la compression de la mémoire vive.
 - **git** : Système de gestion de versions distribué qui sera utile pour installer des paquets plus tard.

3.5

Transfère des optimisations

- `$cp /etc/pacman.conf /mnt/etc/pacman.conf` : Copie la configuration de pacman dans le nouveau système, préservant les optimisations précédemment définies.
- `$cp /etc/pacman.d/mirrorlist /mnt/etc/pacman.d/mirrorlist` : Transfère la liste des miroirs optimisée dans le nouveau système, afin de conserver la liste de miroirs optimisés.

4. Chroot

- `$arch-chroot /mnt`
- La commande `arch-chroot` permet de « rentrer » dans l'environnement du nouveau système installé sur le point de montage `/mnt`.
- Elle modifie la racine du système de fichiers, faisant de `/mnt` le nouveau `/`.
- Une fois à l'intérieur, vous pouvez gérer votre système comme s'il était démarré, par exemple installer des paquets, configurer des services ou éditer des fichiers de configuration, sans avoir à rebooter.

5. Bootloader

5.1 Étapes du processus de démarrage

1. **Firmware** : Le firmware (UEFI) initialise le matériel minimal, localise la partition système EFI (ESP) et charge le bootloader se trouvant dessus.
2. **Bootloader** : Le bootloader (systemd-boot) sélectionne le noyau et charge également l'initramfs en mémoire, puis passe la main au noyau en lui fournissant les paramètres nécessaires.
3. **Initramfs** : Chargé avant l'exécution complète du noyau, l'initramfs sert de système de fichiers temporaire. Il contient les pilotes et outils indispensables pour trouver et monter la véritable racine du système.
4. **Noyau Linux** : Utilisant l'initramfs, le noyau s'initialise, configure les périphériques, monte le système de fichiers racine, et lance le processus init.
5. **Processus init** : Le premier processus est systemd (PID 1). Il lance les services, monte les partitions complémentaires, configure le réseau, met en place et lance l'environnement utilisateur final.

- `$bootctl install` : Installe le chargeur de démarrage (*bootloader*) systemd-boot sur la partition EFI.
- Créer le fichier `/boot/loader/loader.conf` avec le contenu suivant :

```
default arch.conf
timeout 0
```

- Cette configuration définit le fichier `arch.conf` comme l'entrée par défaut et désactive le délai d'attente (`timeout 0`), ce qui lance immédiatement Arch Linux au démarrage.

5.3 Bootloader

- `$blkid /dev/sdX2 > /boot/loader/entries/arch.conf` : Insère la PARTUUID de la partition racine dans `arch.conf` pour éviter de devoir la retaper manuellement.
- Éditer le fichier `/boot/loader/entries/arch.conf` avec le contenu suivant :

```
title Arch Linux
linux /vmlinuz-linux
initrd /intel-ucode.img
initrd /amd-ucode.img
initrd /initramfs-linux-fallback.img
options root=PARTUUID=<PARTUUID> rw
```

5.4 Bootloader

- **title Arch Linux** : Définit le nom qui apparaîtra dans le menu de démarrage.
- **linux /vmlinuz-linux** : Indique le chemin vers le noyau Linux.
- **initrd /intel-ucode.img** : Charge le microcode du processeur Intel.
- **initrd /amd-ucode.img** : Charge le microcode du processeur AMD. Ces fichiers ne sont appliqués que si le matériel correspondant est utilisé.
- **initrd /initramfs-linux-fallback.img** : Charge une image de système de fichier minimal en mémoire (initramfs) qui fournit les pilotes essentiels au noyau. La version fallback contient tous les pilotes disponibles, contrairement à la version standard qui est générée selon la configuration matérielle de l'ordinateur où a été réalisée l'installation.
- **options root=PARTUUID=<PARTUUID> rw** : Spécifie les options de démarrage du noyau, en particulier la partition racine (root) et l'option rw pour un accès en lecture-écriture.

5.5 Fichier fstab

- `$genfstab -t PARTUUID / > /etc/fstab` : Scanne les partitions montées et crée un fichier `/etc/fstab` en utilisant les **PARTUUID** pour identifier de manière unique chaque partition.
- Le fichier `/etc/fstab` spécifie les partitions à monter, leurs points de montage, les types de systèmes de fichiers, les options de montage, et les paramètres de dump et de vérification.
- Le fichier `/etc/fstab` est utilisé pour indiquer à `systemd` quelles partitions monter et comment procéder.
- **Pourquoi PARTUUID ?**
 - Les noms de périphériques comme `/dev/sda`, `/dev/sdb` peuvent changer d'un démarrage à l'autre, surtout si des supports amovibles sont connectés.
 - Les **PARTUUID** sont des identifiants uniques et persistants assignés à chaque partition, garantissant un montage fiable quel que soit l'ordre de détection des périphériques.

6. Initramfs

- **Ajouter le support USB à l'initramfs** : Pour permettre à l'initramfs de monter une partition racine se trouvant sur une clé USB, éditez le fichier de configuration `/etc/mkinitcpio.conf` comme suit :

```
MODULES=(usbhid xhci_hcd)
```

- Ce sont les modules nécessaires pour s'assurer que les pilotes USB requis sont chargés pendant le processus d'initialisation.
- `$mkinitcpio -P` : Reconstitue l'image initramfs en appliquant les modifications.

7. Compression de la mémoire

7 Compression de la mémoire

- **Configurer Zram pour améliorer la gestion de la mémoire** : Pour activer la compression de la mémoire avec Zram, éditez le fichier de configuration `/etc/systemd/zram-generator.conf` comme suit :

```
[zram0]
zram-size = ram / 2
compression-algorithm = zstd
```

- Ces paramètres réservent la moitié de la RAM physique pour les pages compressées en mémoire, augmentant la capacité de mémoire disponible sans recourir à la pagination sur le disque (*swap*). L'algorithme `zstd` offre un bon équilibre entre vitesse et taux de compression.
- `$systemctl enable systemd-zram-setup@zram0` : Configure le service Zram pour qu'il démarre automatiquement au démarrage.

8. Horloge et fuseau horaire

- `$ln -sf /usr/share/zoneinfo/Canada/Atlantic /etc/localtime` : Configure le fuseau horaire du Canada atlantique.
- `$systemctl enable systemd-timesyncd` : Configure le service de synchronisation périodique de l'horloge fourni par systemd pour qu'il démarre automatiquement au démarrage.

9. Langue du système

9

Langue du système

- **Activer la locale en_CA.UTF-8 pour le système** : Pour configurer la langue du système à l'anglais canadien, modifiez le fichier de configuration `/etc/locale.gen` comme suit :

```
#en_CA.UTF-8 UTF-8    (Décommentez cette ligne)
```

- `$locale-gen` : Génère les locales sélectionnées pour le système.
- **Définir la locale par défaut** : Ensuite, configurez la locale système par défaut dans le fichier `/etc/locale.conf` :

```
LANG=en_CA.UTF-8
```

- Ces paramètres assurent que le système utilise l'anglais canadien comme langue par défaut. Vous pouvez configurer la locale de votre choix (par ex. `fr_CA.UTF-8`).

10. Fichier hosts et hostname

10

Fichier hosts et hostname

- **Définir le nom d'hôte du système** : Pour configurer le nom d'hôte, éditez le fichier `/etc/hostname` et insérez le nom d'hôte de votre choix :

```
<nom_hote_choisi>
```

- **Configurer les entrées locales dans `/etc/hosts`** : Ensuite, mettez à jour `/etc/hosts`. Remplacez `<nom_hote_choisi>` par votre nom d'hôte choisi :

```
127.0.0.1    localhost
127.0.1.1    <nom_hote_choisi>
::1          <nom_hote_choisi>.localdomain <nom_hote_choisi>
```

- Ces entrées permettent de mapper le nom d'hôte à l'adresse locale, facilitant ainsi la résolution locale des noms de domaines.

11. Comptes utilisateurs

11 Comptes utilisateurs

- `$useradd -m -g users <nom_utilisateur>` : Crée un nouveau compte utilisateur. Remplacez `<nom_utilisateur>` par le nom d'utilisateur de votre choix.
- `$usermod -G wheel <nom_utilisateur>` : Ajoute le nouveau compte utilisateur au group `wheel` qui permettra à l'utilisateur d'utiliser la commande `sudo`.
- **Configurer les privilèges `sudo`** : Pour permettre aux membres du groupe `wheel` d'utiliser `sudo`, éditez `/etc/sudoers` comme suit :

```
# %wheel ALL=(ALL) ALL    (Décommentez cette ligne)
```

- `$passwd` : Crée le mot de passe du compte `root`.
- `$passwd <nom_utilisateur>` : Crée le mot de passe de l'utilisateur.

12. Arch User Repository (AUR)

12

Arch User Repository (AUR)

- L'Arch User Repository (AUR) est un dépôt communautaire pour Arch Linux qui permet le partage et l'accès à des paquets créés par les utilisateurs, étendant l'offre logicielle au-delà des dépôts officiels.
- Cela fait d'Arch Linux la distribution en existence avec le plus grand nombre de paquets disponibles, et ce, de manière très accessible par rapport à d'autres distributions (par ex. PPA sur Ubuntu).
- `$su <nom_utilisateur>` : Entrer dans le compte utilisateur non privilégié.
- `$cd` : Naviguer vers le répertoire home de l'utilisateur pour avoir l'accès en écriture.
- `$git clone https://aur.archlinux.org/yay-bin.git` : Récupérer les fichiers nécessaires pour construire Yay, un gestionnaire de paquets pour l'AUR.
- `$cd yay-bin` : Naviguer dans le répertoire contenant les fichiers.
- `$makepkg -sri` : Construire et installer le paquet de Yay.
- `Ctrl` + `D` ou `$exit` : Sortir du compte utilisateur non privilégié.

13. Installation d'un bureau graphique

13

Installation d'un bureau graphique

- ...

14. Redémarrage

- `Ctrl` + `D` ou `$exit` : Sortir du chroot.
- `$umount -R /mnt` : Démonter les partitions `/mnt/boot` et `/mnt`.
- `$reboot` : Redémarrer le système.

Eric Hervet et Andy Couturier

Université de Moncton, 27 novembre 2024

